**ALU Design Project: Design Review 2**

**Due: 04/19/2013**

**Digital Integrated Circuits - Spring 2013**

**DIC-sie Chicks:**

Chaitanya Bhave, Evelyn Ponce, and Sam Schulman

**Progress and Description:**

In order to be ready for the second design review, team DIC-sie Chicks had to create an 8-bit adder, an 8-bit 2’s complement, and an 8-bit register. Since the group has three members, the shifter was made the arbitrary function, and therefore will not be done until the final review. The 8-bit adder was created from 8 one-bit full adders, with each carry-out bit connected to the next adder’s carry-in bit. The one-bit adder consists of AND, OR, and XOR gates, in which the XOR gate was constructed from previously built NAND gates. The 2’s complement was put together using the 8-bit inverter and the 8-bit adder, where the A input was inverted and passed to the A input of the adder, while the B input of the adder was grounded, and the carry-in was set to high. The register was created using an inverted version of the circuit for the negated-edge-sensitive D-flip-flop master-slave register schematic used for the most recent homework set.

These components had to combined to create the ALU, which is now complete, with the exception of the shifter, which will serve as the arbitrary function (given that our group is a 3-person group and is thus not otherwise required to include the shifter in the project). For this design review, the Logisim simulation was also updated to reflect the newly-chosen arbitrary function.

In order to minimize testing issues, all required functions were generated in 1-bit versions fist. These smaller versions were then tested and iterated to create the 8-bit components required for the ALU. As a result, it was simpler to identify and correct errors in the base schematics. This also allowed for greater ease in observing the behavior of the 1-bit pieces of each function, which provided another method of ensuring the functions were actually serving their intended purpose.

**Work Breakdown Schedule:**

By 4/23/2013 - Create and test 8-bit Shifter, fix errors from previous Design Reviews (if possible)

By 4/25/2013 - Work out W/L values and metric math

By 4/29/2013 - Final touches, complete presentation

**Delays:**

We used the simulations labeled below to get the worst-case delay for each operation.

|  |  |  |
| --- | --- | --- |
| **Operation** | **Worst-Case Delay** | **Scenario** |
| ADD | 465.8 ps | B=00000001, “high” and A= goes “low” to” high”; 00000000 to 11111111 and carry in value has to be low |
| 2COMP | 127 ps | When all the bits are going from low to high; when A transitions from 00000000 to 11111111 |
| AND | 314.429 ps | B= 11111111; “high” and A= transitions from “low” to “high”; 00000000 to 11111111 |
| OR | 327.1 ps | A & B=00000000 to 11111111; goes from “low” to “high” |
| Pass A | 534.22 ps | A goes from “low” to “high”; 00000000 to 11111111 |

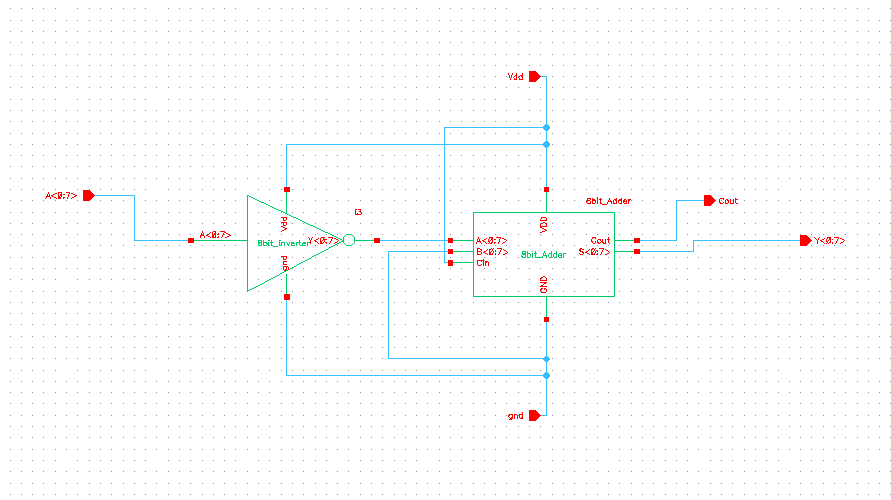
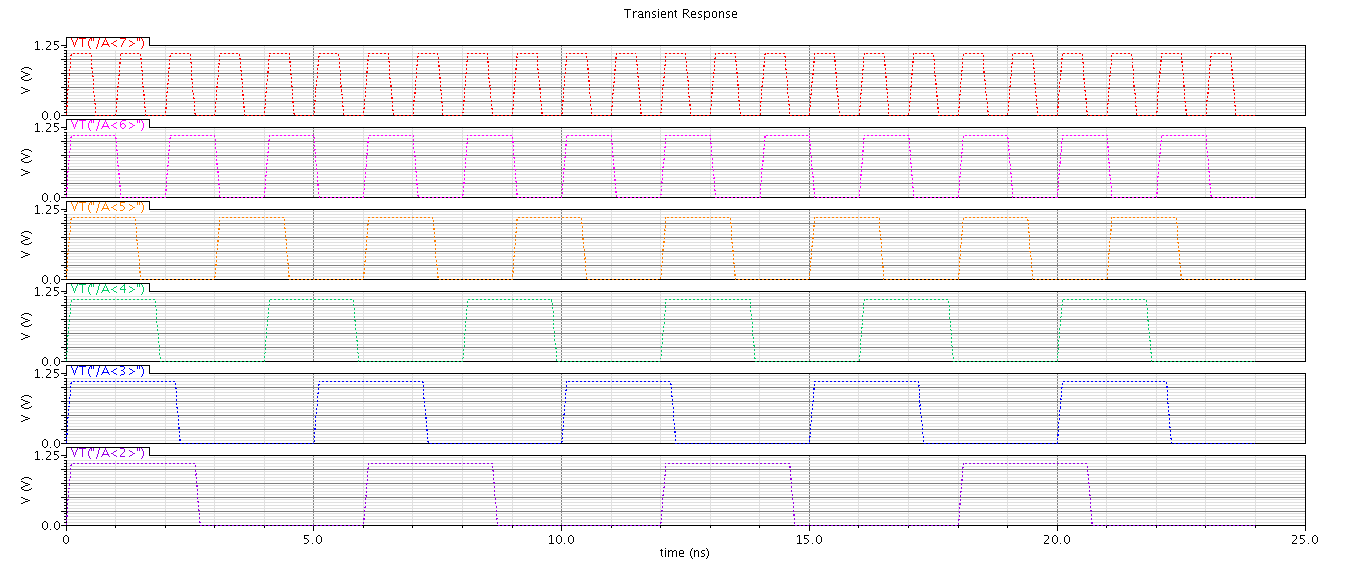
2’s Complements: 

Figure 1: This is the 8 bit two’s complements we created for the ALU.

****

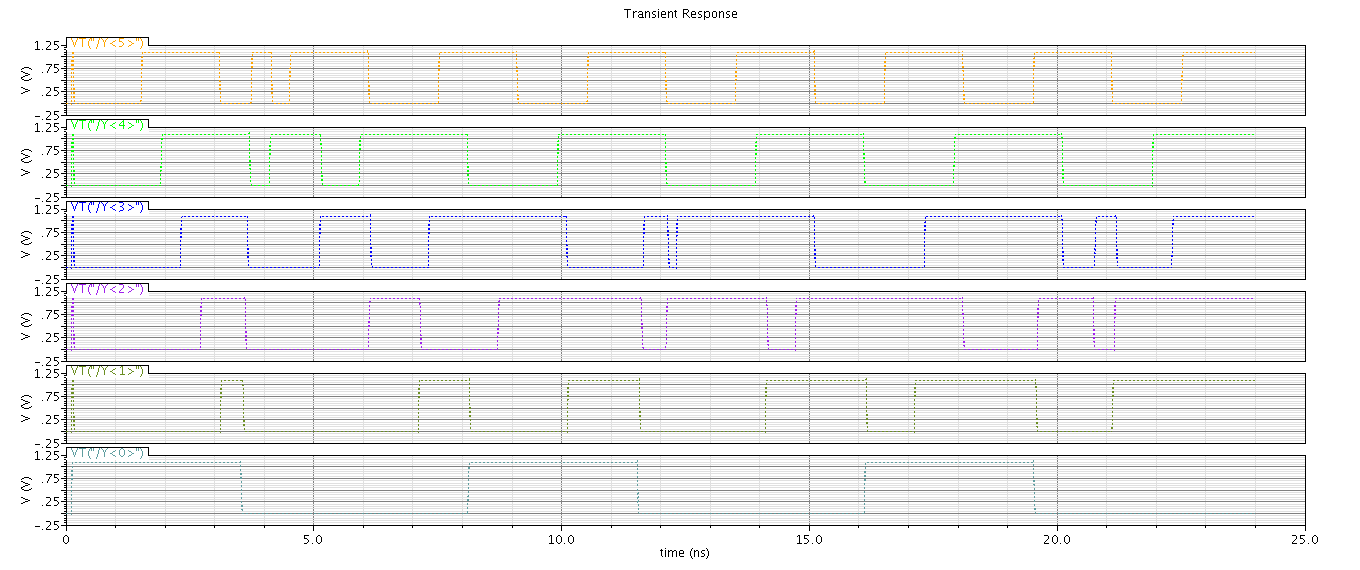
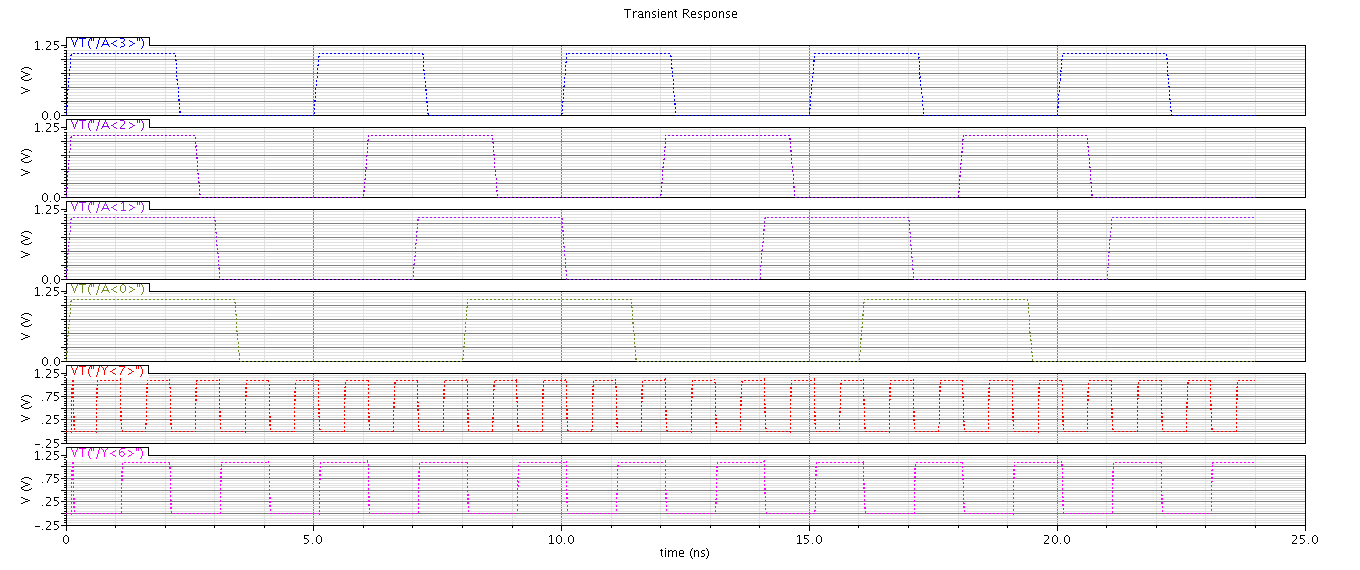


Figure 2,3,4: Are the simulations that correspond for the 8 bit two’s complement for each bit. We used these simulations to calculate the worst case delay and the scenario that corresponds to it.

Adder:

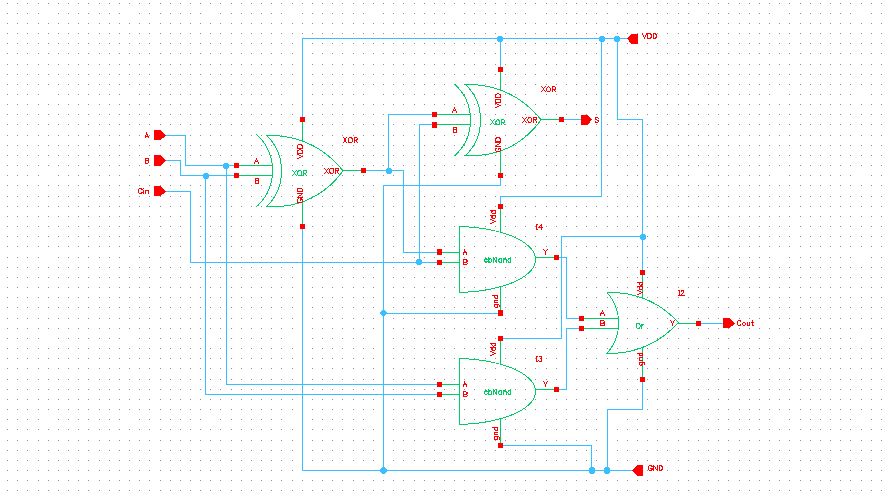


Figure 5: This is the basic full adder that we created that we would use to build the 8 bit adder and the two’s complements.

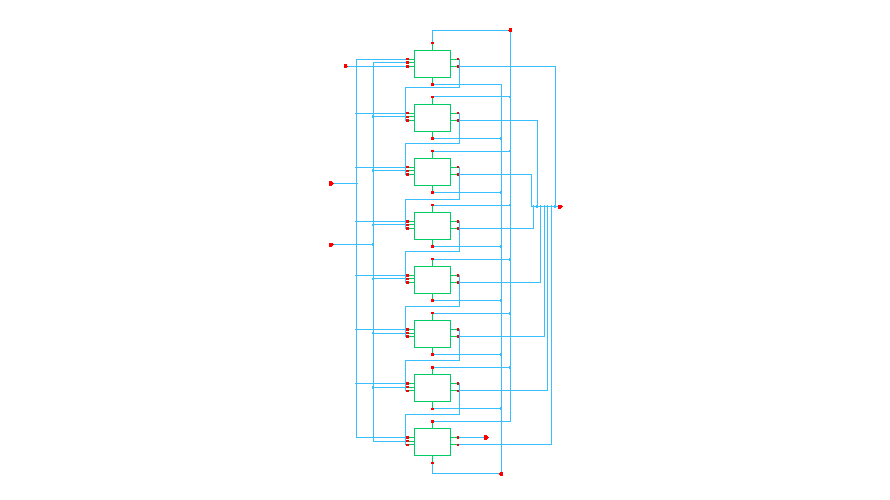


Figure6: The 8 bit adder that we created using the basic full adder we previously created.

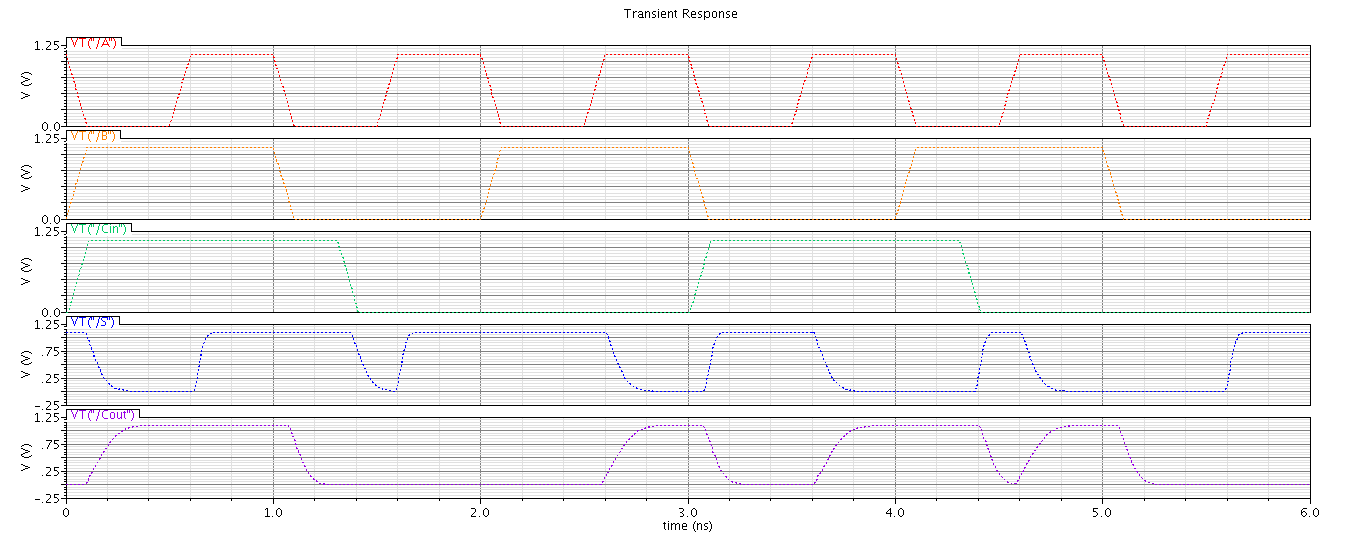


Figure 7: Simulation for the basic full adder we created. We used these simulations to calculate the worst case delay and the scenario that corresponds to it.

And:

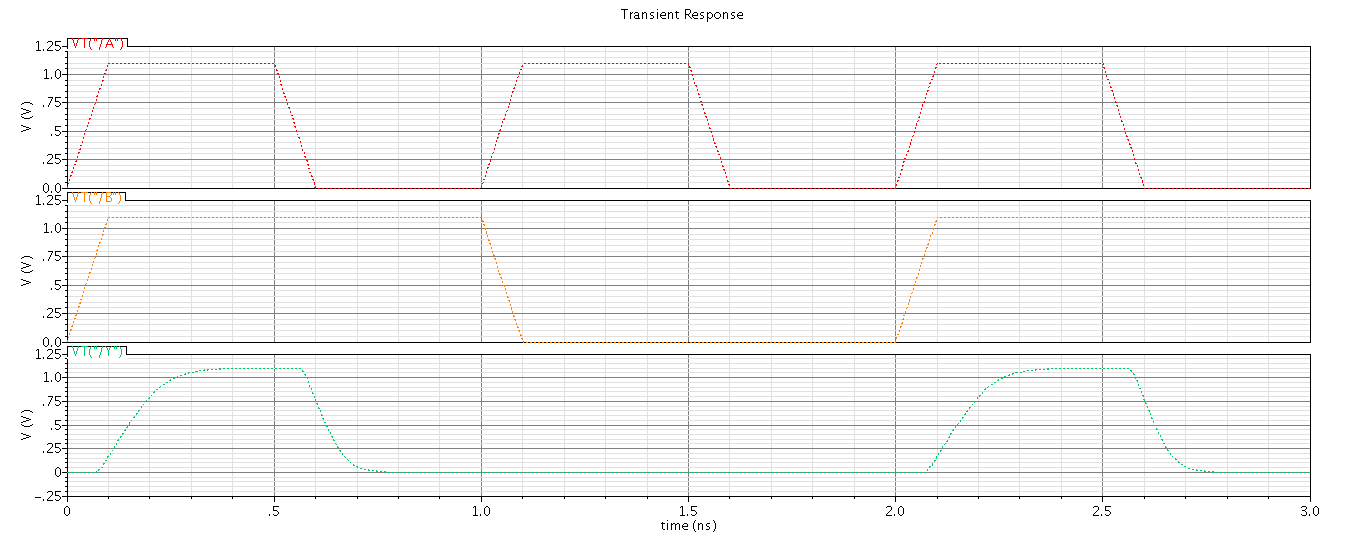


Figure 8: Simulation for the basic And component we created that we used to calculate the worst case delay and the scenario that corresponds to it.

Or:

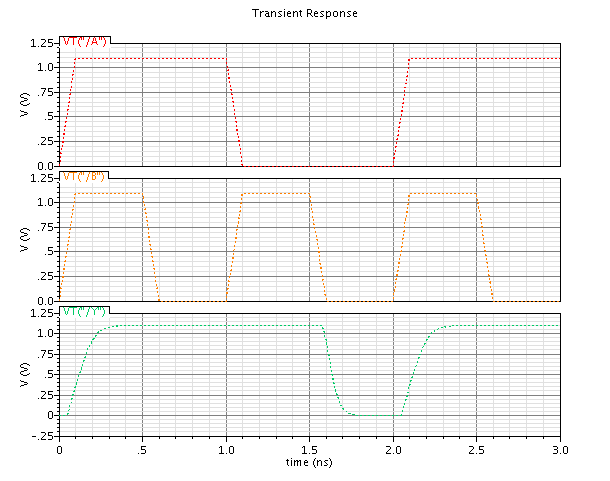


Figure 9: Simulation for the basic Or component we created that we used to calculate the worst case delay and the scenario that corresponds to it.

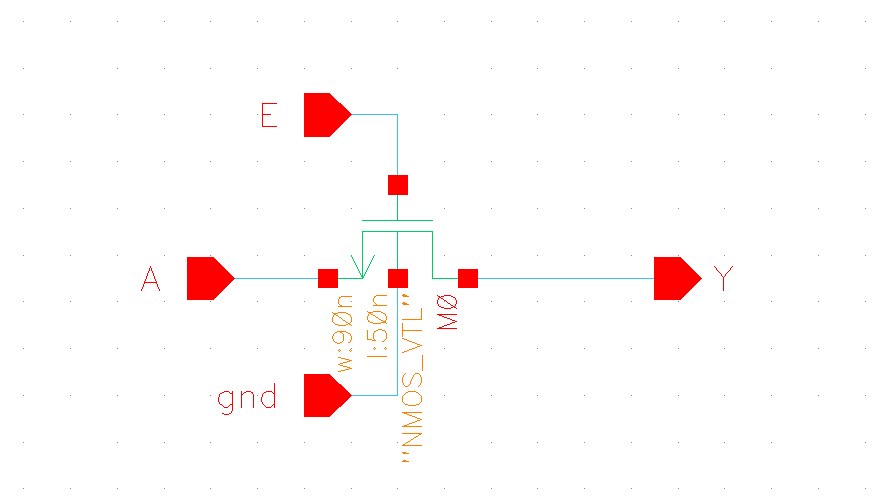
PassA: 

Figure 10:This the schematic of the basic pass A component.

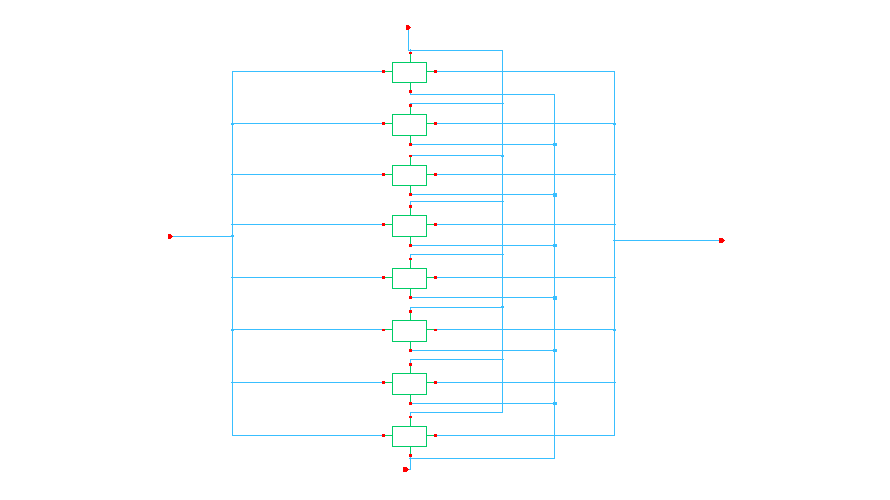


Figure 11: This is the schematic of the 8 bit pass A component we created using the basic pass A component.

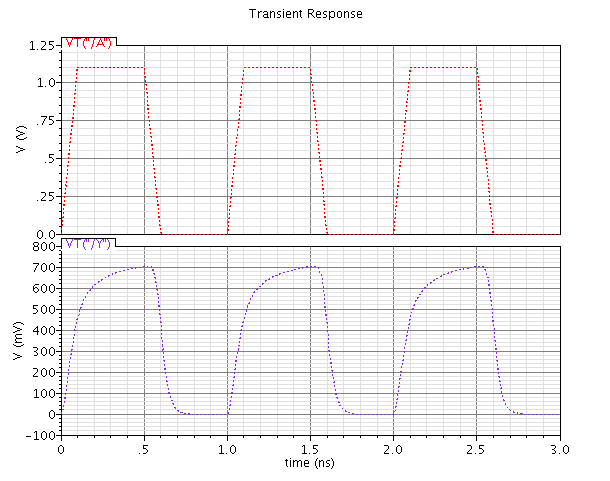


Figure 12: Simulations for the basic pass A component that we used to calculate the worst case delay and the scenario associated with it.

Register:

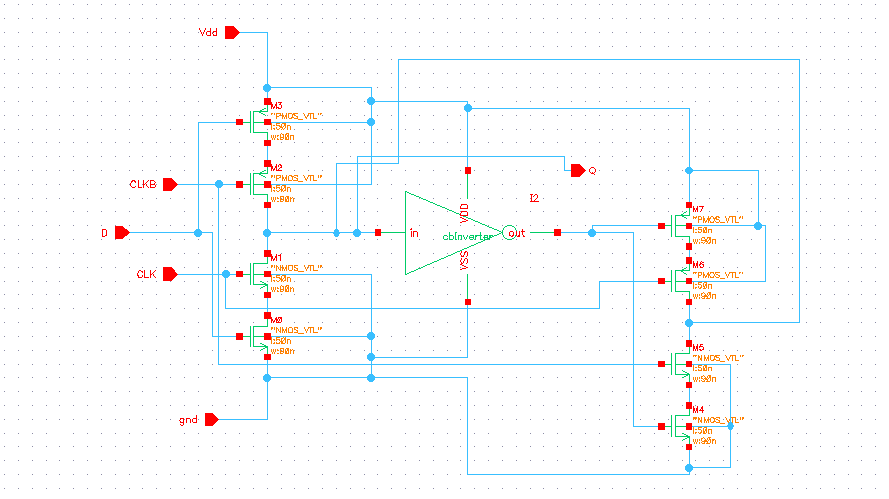


Figure 13: This is the base register for the master-slave register we used in previous homework as SE1.

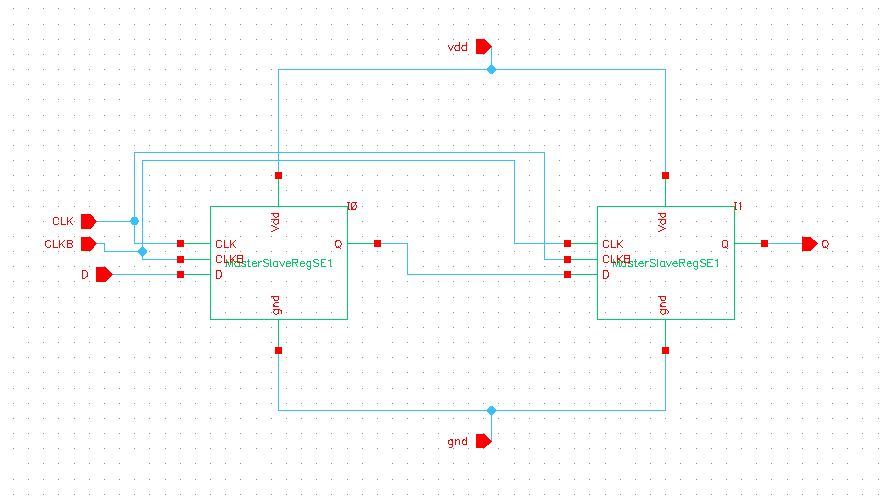


Figure 14: The schematic above is the master-slave register that was SE2 in the previous homework.

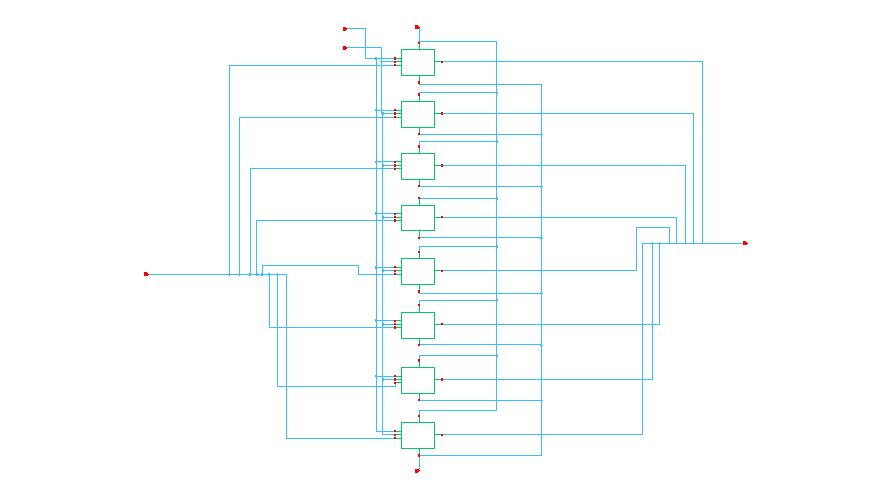


Figure 15: We used the master-slave registers we created to build an 8 bit register.

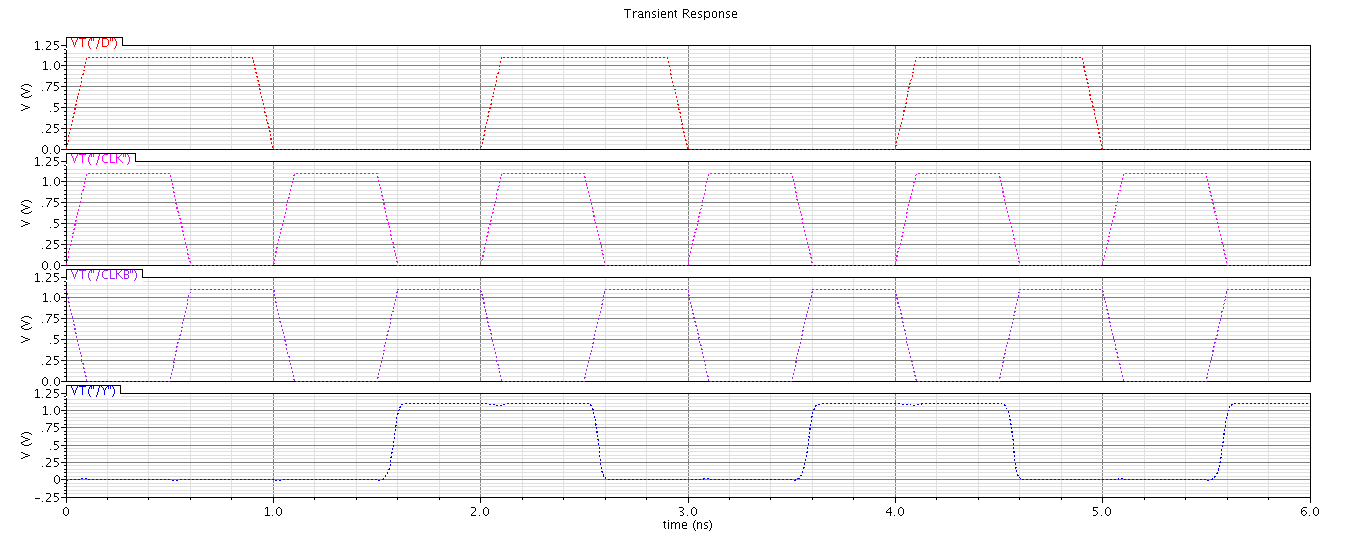


Figure 16: This the simulation for the master-slave register, SE2, that we used to get the worst-case delay and scenario associated with it.

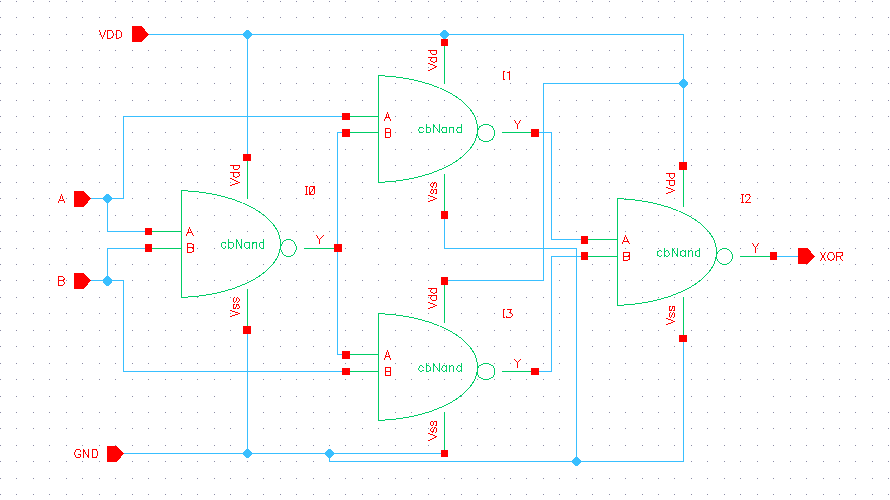
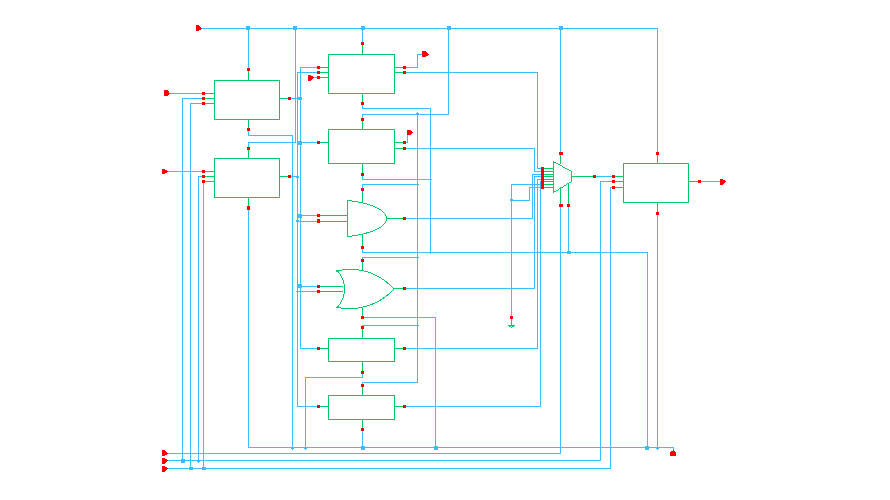
XOR:

Figure 17: We created the XOR gate since this component was need to build the full adder that was used in the 8 bit adder and 8 bit two’s component.

ALU:



MUX

Or

And

NOP

Pass A

2’s

Complement

Adder

Register

Register

Register

Figure 18: ALU we created in cadence that combines all the components we created.

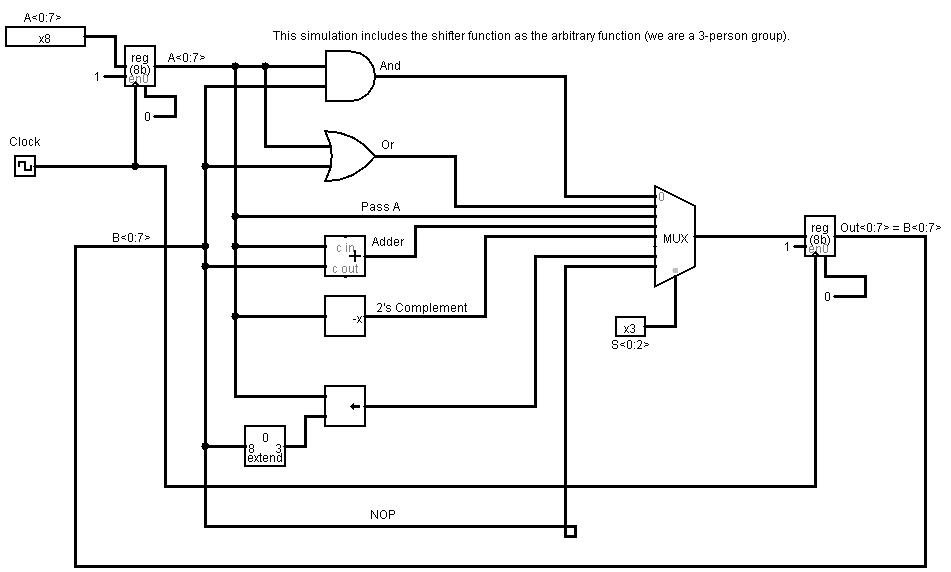


Figure 19: This is the final ALU that we will create in Logisim. This ALU schematic includes our arbitrary function of a shifter.